Innovate 2011 DevCamp

Magento Workshop 1

Customizing the Onepage Checkout

Vinai Kopp

Setup and Configuration

The workshop is designed to be followed on a Magento CE 1.6 Installation. For Innovate attendees a VirtualBox image is available that contains a preconfigured environment.
If you prefer to use your own development environment, you may install the TGZ archive of the source and the DB dump.

The code for the hands-on labs is available in pre-baked snipplets in a separate archive file for your convenience.

Workshop Resources:

A. VirtualBox VM Image (includes all workshop assets)
Magento_Workshop.vdi
B. Magento Workshop Files Archive (Magento, the pre-baked code and the DB-dump)
Magento_Workshop_Full.tgz
C. Archive containing only the pre-baked code snipplets
Magento_Workshop1_Code.tgz

Preparations with the Innovate VirtualBox Image:

1. Fire up the VirtualBox VM using the supplied disk image
2. Open the Firefox Browser (a shortcut button is on the desktop)
3. If the Demo Magento homepage doesn't load automatically please visit http://workshop.dev/
4. Open up the NetBeans IDE (there is a shortcut on the VM Desktop and in the toolbar)

Preparations without the Innovate VirtualBox Image:

1. Unpack the archive *Magento_Workshop_Full.tgz*.
2. Create a database and import the database dump *Magento_Workshop.sql* found in the archive.
3. Execute the SQL, replacing "your-domain.dev" with whatever your development domain is: `UPDATE core_config_data SET value="http://your-domain.dev/" WHERE path LIKE '%base_url';`
4. Edit the settings in the file *magento/app/etc/local.xml* file to point to the imported database.
5. Visit the homepage of the Magento Installation to confirm all is working
6. Open the Magento Installation in the IDE of your choice
7. Log into the admin panel and navigate to "**System** > **Configuration**". Select "**Shipping Method**" on the left and make sure only one shipping method is enabled.

VirtualBox Image Authentication Credentials:

If you are using the VirtualBox VM you can use the following credentials:

Magento
Magento Frontend URL: http://workshop.dev/
Magento Admin URL: http://workshop.dev/admin
Magento Admin User: admin
Magento Admin Password: magento123

MySQL
PhpMyAdmin URL: http://workshop.dev/phpmyadmin
MySQL Root User: root
MySQL Root Password: magento123

Linux
VM User: magento
VM Password: magento123

Directory Structure

In the VirtualBox Image all files can be found at */var/www/workshop*. A project has already been created for you in NetBeans.
Inside the workshop directory you have the following sub directories:

/magento
The Magento Application Code as well as the example modules

/onepage-checkout-workshop/Code
The pre-baked Code Snipplets for the hands-on Labs

/onepage-checkout-workshop/Lab N
The Code as it is before the specified hands-on Lab begins

/onepage-checkout-workshop/Lab N finished
The Code as it is after the specified hands-on Lab

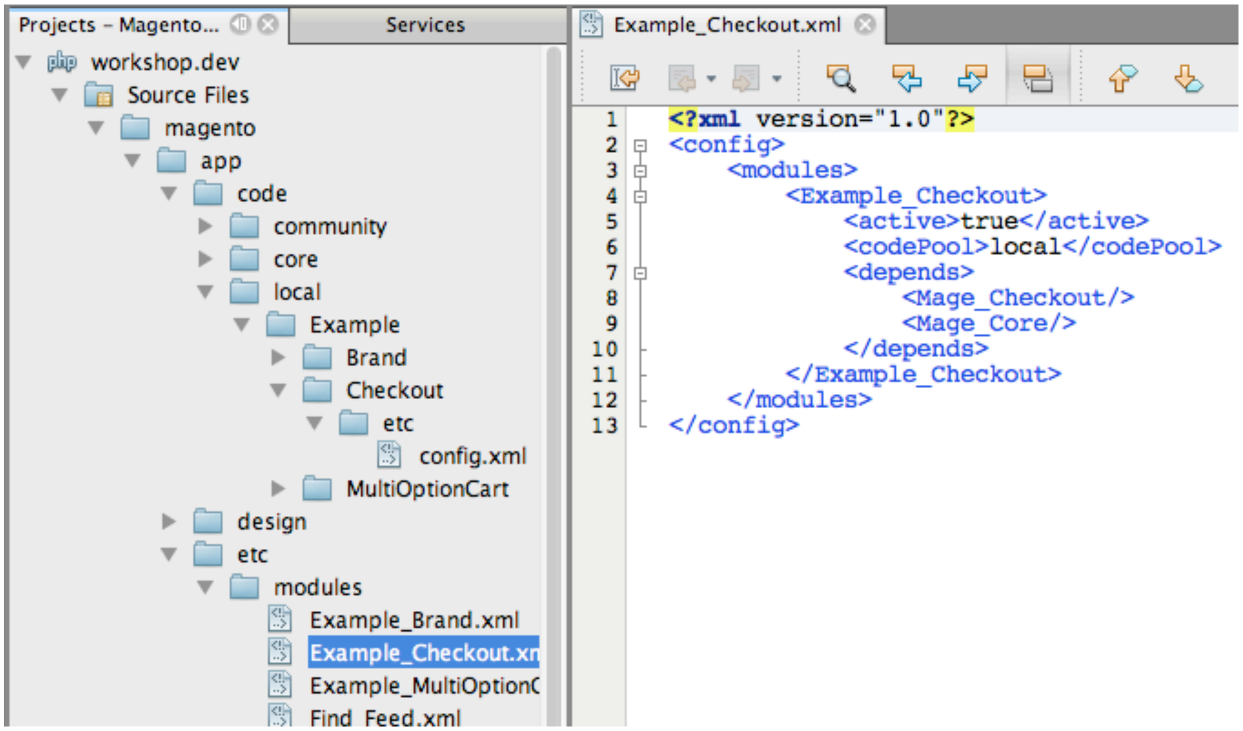
/performance-workshop/
The Code for the other Innovate DevCamp Magento Workshop

Lab 01 - Create an empty Module

1. Open the IDE project
2. Create the file *magento/app/etc/modules/Example_Checkout.xml* and add the code block 01-01 as the contents of that file.

Code Block 01-01:

```
<?xml version="1.0"?>
<config>
      <modules>
            <Example_Checkout>
                  <active>true</active>
                  <codePool>local</codePool>
                  <depends>
                        <Mage_Checkout/>
                        <Mage_Core/>
                  </depends>
            </Example_Checkout>
      </modules>
</config>
```



```
<?xml version="1.0"?>
<config>
    <modules>
        <Example_Checkout>
            <active>true</active>
            <codePool>local</codePool>
            <depends>
                <Mage_Checkout/>
                <Mage_Core/>
            </depends>
        </Example_Checkout>
    </modules>
</config>
```

3. Create the directory *magento/app/code/local/Example/Checkout/etc*. Please note the **capitalization** of the **directory names**.

4. Create the file *magento/app/code/local/Example/Checkout/etc/config.xml* and add the code block 01-02 as the contents of the file.

Code 01-02:

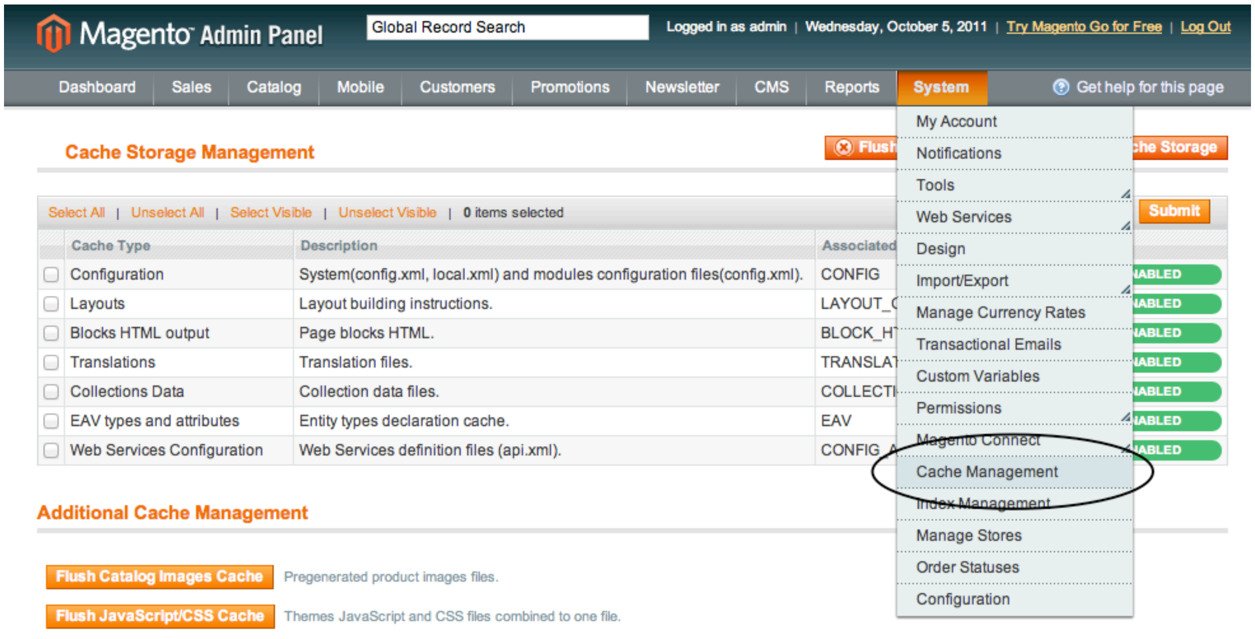
```
<?xml version="1.0"?>
<config>

</config>
```

Test the Code

Since the module doesn't do anything yet, we can only check if Magento recognizes it exists. Every time you make a change to a configuration file you must refresh the configuration cache to make it take effect.

Clear the Cache so Magento "sees" our module

1. Log in to the admin panel at *http://workshop.dev/admin*
2. Navigate to "**System** > **Cache Management**"



Cache Storage Management

Select All | Unselect All | Select Visible | Unselect Visible | 0 items selected

	Cache Type	Description	Associated	
☐	Configuration	System(config.xml, local.xml) and modules configuration files(config.xml).	CONFIG	ABLED
☐	Layouts	Layout building instructions.	LAYOUT_(	ABLED
☐	Blocks HTML output	Page blocks HTML.	BLOCK_HT	ABLED
☐	Translations	Translation files.	TRANSLAT	ABLED
☐	Collections Data	Collection data files.	COLLECTI	ABLED
☐	EAV types and attributes	Entity types declaration cache.	EAV	ABLED
☐	Web Services Configuration	Web Services definition files (api.xml).	CONFIG_A	ABLED

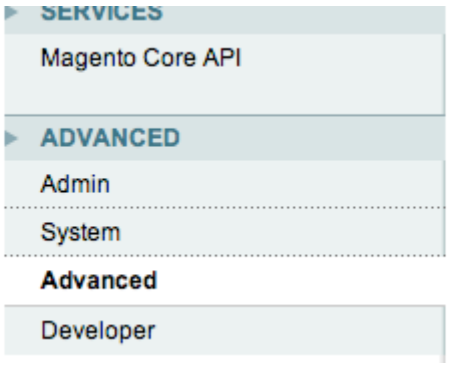
Additional Cache Management

Flush Catalog Images Cache — Pregenerated product images files.

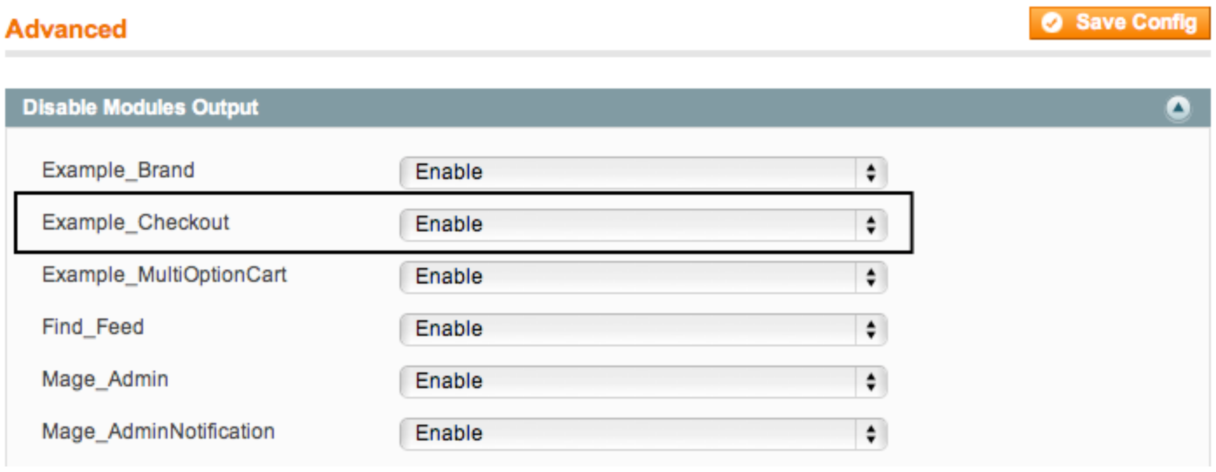
Flush JavaScript/CSS Cache — Themes JavaScript and CSS files combined to one file.

3. Select the checkbox in the top row marked **Configuration**
4. Click the **Submit** button to refresh the configuration cache

5. Navigate to the page “**System** > **Configuration**”
6. Select the section “**Advanced**” on the bottom left



7. Click on the blue title panel labeled “**Disable Module Output**” to expand it
8. Check our module **Example_Checkout** is listed



Example_Brand	Enable
Example_Checkout	Enable
Example_MultiOptionCart	Enable
Find_Feed	Enable
Mage_Admin	Enable
Mage_AdminNotification	Enable

Since Example_Checkout is listed it means Magento is picking up our configuration.

Lab 02 - Add Code to hook in the Checkout

1. Add the bold code from block 02-01 inside the `<config>` node in the *magento/app/code/local/Example/Checkout/etc/config.xml* file.
   Please note, there is **no whitespace** before or after the **text contents** of the `<Example_Checkout>` node.

Code 02-01:

```
<?xml version="1.0"?>
<config>
  <frontend>
    <routers>
      <checkout>
        <args>
          <modules>
            <Example_Checkout
   before="Mage_Checkout">Example_Checkout</Example_Checkout>
          </modules>
        </args>
      </checkout>
    </routers>
  </frontend>
</config>
```

2. Create the directory *magento/app/code/local/Example/Checkout/controllers*
   Note the directory name "**controllers**" is plural and in all **lower case**.
3. Create the file
   magento/app/code/local/Example/Checkout/controllers/OnepageController.php
   and add the code block 02-02 as the contents.

Code 02-02:

```
<?php

require_once 'Mage/Checkout/controllers/OnepageController.php';
class Example_Checkout_OnepageController
      extends Mage_Checkout_OnepageController
{
}
```

Lab 03 - Create Methods to add our logic

1. Open the file *magento/app/code/local/Example/Checkout/controllers/OnepageController.php*
2. Add the code block 03-01 into the class `Example_Checkout_OnepageController`.

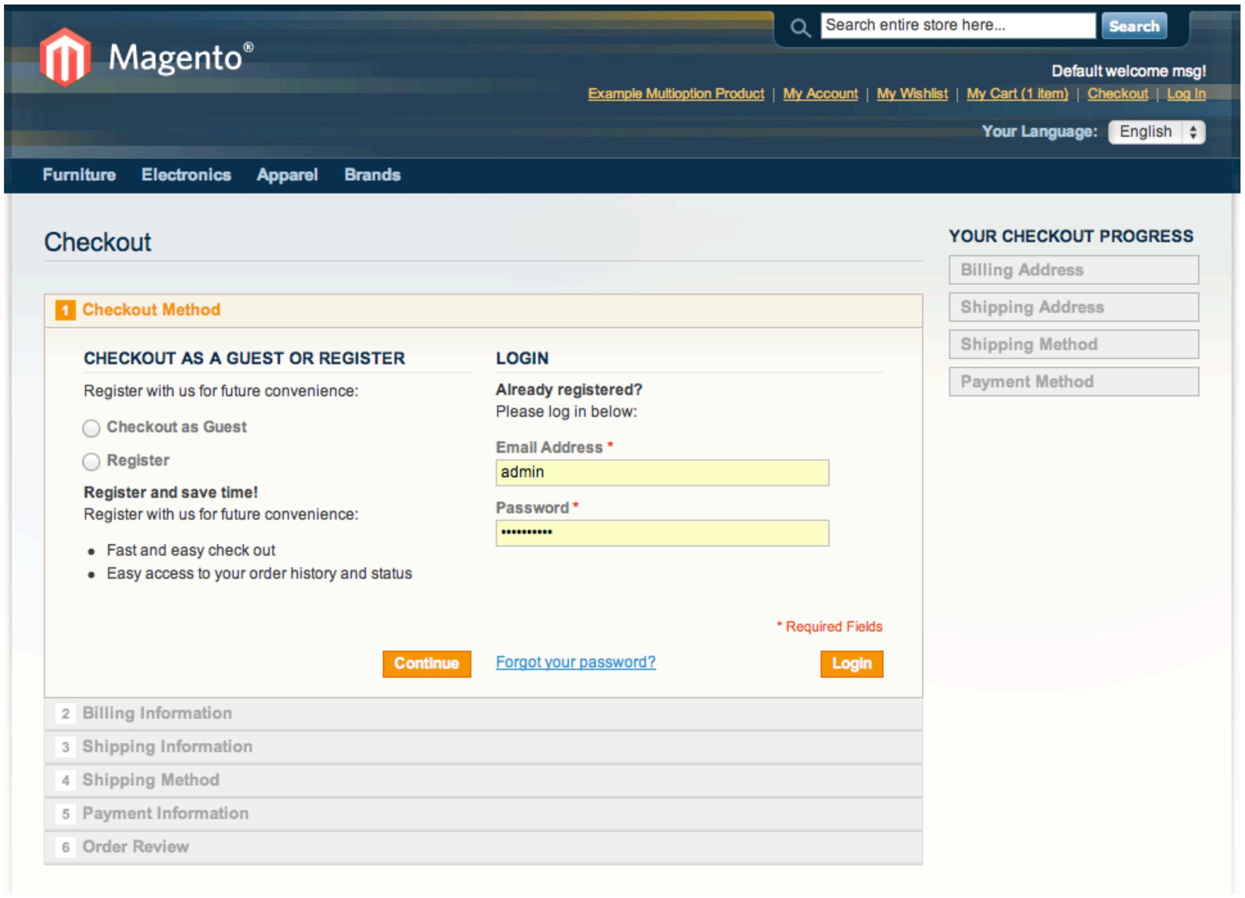
Code 03-01:

```
public function saveBillingAction()
{
  parent::saveBillingAction();
  $this->_checkSkipShippingMethod();
}

public function saveShippingAction()
{
  parent::saveShippingAction();
  $this->_checkSkipShippingMethod();
}

protected function _checkSkipShippingMethod()
{
  $helper = Mage::helper('core');

  // Check the session hasn't expired
  $body = $this->getResponse()->getBody();
  if ($body)
  {
    $result = $helper->jsonDecode($this->getResponse()->getBody());
    if (array_key_exists('goto_section', (array)$result)
        && 'shipping_method' === $result['goto_section']
    )
    {
    }
  }
}
```

Test the Code

Currently we can only test the module by verifying it does not trigger an exception.

1. Clear the configuration cache (see Lab 1 for detailed instructions).
2. On the front end, add a product to the cart and proceed to the checkout.
3. If the checkout loads without errors your code is working.



Magento®
Search entire store here...
Search
Default welcome msg!
Example Multioption Product | My Account | My Wishlist | My Cart (1 item) | Checkout | Log In
Your Language: English
Furniture
Electronics
Apparel
Brands

Checkout

1 Checkout Method

CHECKOUT AS A GUEST OR REGISTER

Register with us for future convenience:

Checkout as Guest

Register

Register and save time!

Register with us for future convenience:

- Fast and easy check out
- Easy access to your order history and status

Continue

LOGIN

Already registered?

Please log in below:

Email Address *

admin

Password *

••••••••••

* Required Fields

Forgot your password?

Login

2 Billing Information

3 Shipping Information

4 Shipping Method

5 Payment Information

6 Order Review

YOUR CHECKOUT PROGRESS

Billing Address

Shipping Address

Shipping Method

Payment Method

Lab 04 - Check the available Shipping Methods

1 Open the file *magento/app/code/local/Example/Checkout/controllers/OnepageController.php*

2. Add the code from the block 04-01 into the inner `if()` section of the method `_checkSkipShippingMethod()`

Code 04-01:

```
$onepage = $this->getOnepage();
$shippingMethods = $this->_getAvailableShippingMethods($onepage);
if (count($shippingMethods) === 1)
{
}
```

3. Add the new method `_getAvailableShippingMethods()` from code block 04-02 into the class `Example_Checkout_OnepageController`.

Code 04-02:

```
protected function _getAvailableShippingMethods(
                           Mage_Checkout_Model_Type_Onepage $onepage)
{
      $address = $onepage->getQuote()->getShippingAddress();
      $methods = array();
      foreach ($address->getGroupedAllShippingRates() as $carrier)
      {
            foreach ($carrier as $rate)
            {
                  $methods[] = $rate->getCode();
            }
      }
      return $methods;
}
```

Lab 05 - Select the Shipping Method and Continue

1. Add the *c*ode block 05-01 into the inner `if()` block in the method `_checkSkipShippingMethod()` in the file *magento/app/code/local/Example/Checkout/controllers/OnepageController.php*

Code 05-01:

```
$onepage->getQuote()->setTotalsCollectedFlag(false);
$onepage->saveShippingMethod($shippingMethods[0]);

$this->getLayout()->getUpdate()->setCacheId(null);

$result['goto_section'] = 'payment';
$result['update_section'] = array(
      'name' => 'payment-method',
      'html' => $this->_getPaymentMethodsHtml()
);
$result['allow_sections'] = array('shipping', 'shipping_method');

$onepage->getCheckout()
      ->setStepData('shipping_method', 'complete', true);

$this->getResponse()->setBody($helper->jsonEncode($result));
```

Test the Code

Now the module is complete we can check everything works as planned.

1. If you haven’t done so yet, add a product to the cart and proceed to the checkout.
2. Select **Checkout as a Guest** or **Register**
3. Fill the **Billing Information** form. Select the “**Ship to this address**” option below the form and click the **Continue** button.
4. Check the next step is the **Payment Information** step and in the progress bar on the right **Flat Rate** is selected as the current shipping method.
   If so the `saveBillingAction()` method works as expected.
5. Now check the `saveShippingAction()` method. Click on the header of the **Shipping Information** step to back up to that step.
6. Click the **Continue** button. Once again the Shipping Method step should be skipped and the **Payment Information** step should be visible.